School of Computer Engineering & Technology

COMPUTER VISION

MINI PROJECT REPORT

List of Team members(with PRN and Roll no):

GAUTAM DWIVEDI (1032220982)
MUNI REDDY (1032211545)
UPKAR CHATTA (1032211292)
ROHAN YADAV (1032211239)

Problem Statement: PARKING MANAGEMENT SYSTEM USING YOLO(V3)

Introduction :-

In an increasingly urbanized world, managing parking efficiently has become a critical aspect of urban infrastructure. Traditional parking management systems often struggle to keep up with the growing demand and complexities of modern cities. However, with the advent of advanced technologies like deep learning and computer vision, new avenues for tackling parking management challenges have emerged.
One such technology making waves in the realm of parking management is YOLO (You Only Look Once), particularly its third version, YOLOv3. YOLOv3 is a state-of-the-art object detection algorithm that excels in real-time detection of objects within images and videos. By leveraging its capabilities, parking management systems can achieve a higher level of accuracy and efficiency in detecting and monitoring parking spaces.

Objectives :-

1. **Scalability and Adaptability**: Develop a scalable and adaptable parking management solution that can be deployed across various parking facilities, including outdoor parking lots, indoor garages, and multi-level parking structures.
2. **Real-time Detection and Monitoring**: Implement a system capable of real-time detection and monitoring of parking spaces and vehicles within parking lots using the YOLOv3 object detection algorithm.
3. **Enhanced User Experience**: Improve the overall parking experience for drivers by guiding them to available parking spaces efficiently through mobile apps, electronic signage, or other communication channels integrated into the system.

Methodology :-

Methodology for Implementing the Parking Management System Using YOLOv3:

1. **Requirement Analysis:**
   - Conduct a comprehensive analysis of the requirements and objectives of the parking management system, considering factors such as parking lot size, expected traffic volume, types of vehicles, and desired features.
2. **Data Collection and Preparation:**

- Gather a diverse dataset of images and videos containing parking lots with varying lighting conditions, vehicle types, and occupancy levels.
- Annotate the dataset to mark the locations of parking spaces and vehicles within the images, ensuring accuracy and consistency for training the YOLOv3 model.

3. **Model Training:**
   - Preprocess the annotated dataset to prepare it for training, including data augmentation techniques such as rotation, scaling, and flipping to increase dataset diversity.
   - Train the YOLOv3 model using the prepared dataset, adjusting hyperparameters as needed to optimize performance for parking space and vehicle detection.
4. **System Integration:**
   - Integrate the trained YOLOv3 model into the parking management system, developing software modules for real-time object detection and monitoring.
   - Connect the system to surveillance cameras deployed in parking lots to capture live video feeds for analysis.
5. **Real-time Object Detection:**
   - Implement algorithms for real-time object detection using the YOLOv3 model, processing video streams from surveillance cameras to identify parking spaces and vehicles.
   - Develop mechanisms for tracking vehicles as they enter and exit parking spaces, updating the occupancy status of each space in real-time.
6. **User Interface Development:**
   - Design and develop user interfaces for interacting with the parking management system, including web-based dashboards, mobile applications, and electronic signage displays.
   - Implement features for displaying real-time parking availability information to drivers and administrators, with intuitive controls for navigating and managing parking resources.
7. **Data Analysis and Insights:**
   - Collect and store data generated by the parking management system, including parking occupancy rates, usage patterns, and historical trends.
   - Analyze the data to extract actionable insights for optimizing parking operations, improving user experience, and informing future decision-making.
8. **Testing and Validation:**
   - Conduct thorough testing of the parking management system to validate its functionality, accuracy, and performance under various conditions.
   - Perform usability testing with end-users to gather feedback and identify areas for improvement.
9. **Deployment and Maintenance:**
   - Deploy the parking management system in target parking facilities, ensuring proper installation, configuration, and integration with existing infrastructure.
   - Establish procedures for ongoing maintenance, monitoring system performance, applying updates, and addressing any issues or concerns that arise.

10. **Training and Support:**

- Provide training and support to users and administrators of the parking management system, offering guidance on system operation, troubleshooting, and best practices for maximizing benefits.

Literature survey:-

(1) The paper utilizes the YOLO algorithm for object detection and tracking.The YOLO algorithm is integrated with a novel enhanced vehicle parking mechanism to improve parking space finding time and efficiency.The research work aimed to design and develop a novel enhanced vehicle parking mechanism using the YOLO algorithm, focusing on parameters like parking space finding time and parking efficiency.The proposed architecture successfully improved parking effectiveness and reduced parking space search time, enhancing the overall parking experience for drivers.

(2) The paper implements an automatic vehicle license plate recognition system based on the YOLO v4 algorithm, combining LINE Bot and Line Notify for pushing parking fee messages.The YOLOv4 algorithm is utilized for license plate recognition, with a focus on license plate positioning and character identification.The algorithm structure consists of components like Conv, Bn, and Mish activation functions, as well as Conv, Bn, and Leakyrelu activation functions.

(3) The paper utilizes Automatic Number Plate Recognition (ANPR) with OCR using the Raspberry Pi camera for license plate detection.It employs the Yolo image detection algorithm for fast recognition and a combination of Yolov4 and Contour detection techniques.The research improved OCR accuracy by using Convolutional Neural Network (CNN) instead of Tesseract and utilized a combination of Yolo and contour detection for image detection The research significantly improved OCR accuracy by using CNN instead of Tesseract, enhancing image detection capabilities.The combination of Yolov4 and Contour detection techniques contributed to faster image recognition.

(4) The paper implements a parking slot detection system using You Only Look Once version 8 (YOLOv8) for real-time monitoring.The methodology involves training the YOLOv8 algorithm on a dataset to learn features, generate critical weights, and analyze video footage to distinguish between empty and occupied parking slots.The proposed method customizes and fine-tunes the YOLOv8 model for parking spot identification using annotated pictures of parking lots with defined bounding boxes and class labels.The YOLOv8 model demonstrated flexibility, resilience, excellent processing speed, higher accuracy, real-time performance, and a high degree of flexibility in handling dynamic parking situations.

Code:-

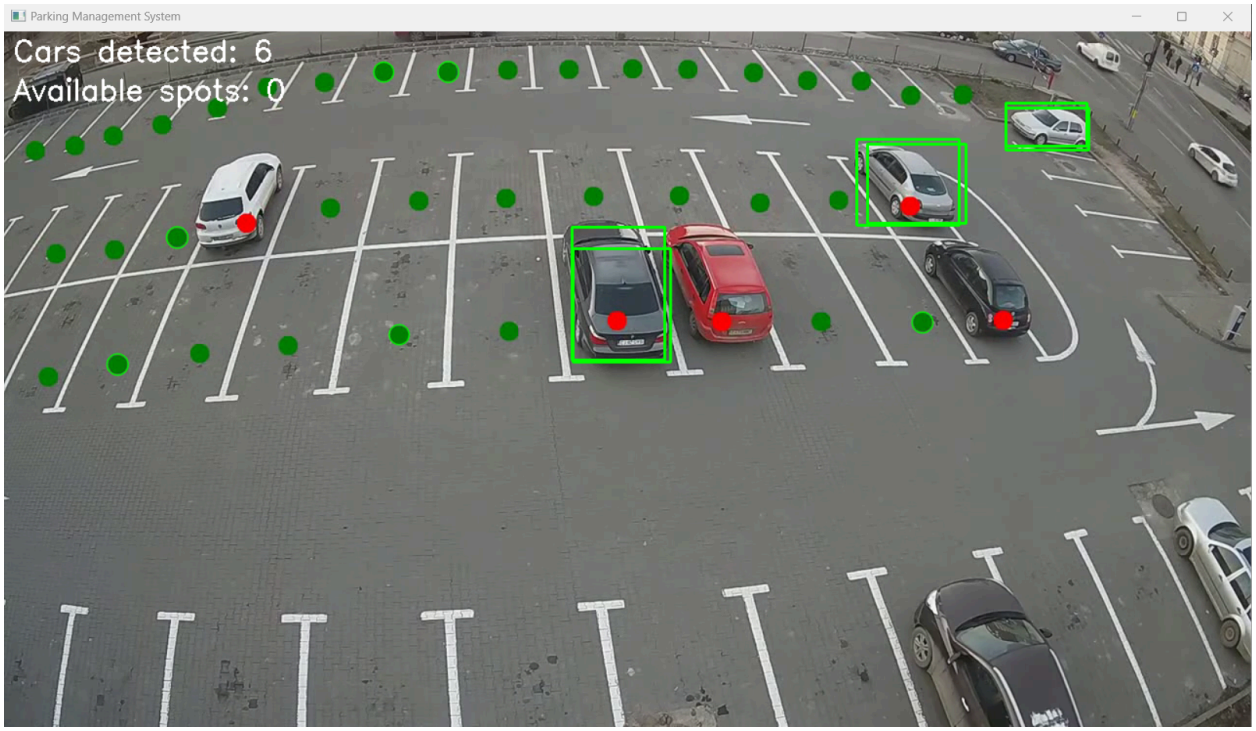
In [3]:
```
net = cv2.dnn.readNet("yolov3 (1).weights", "yolov3.cfg")
```

In [12]:
```
import cv2
import numpy as np

# Load YOLO
net = cv2.dnn.readNet("yolov3 (1).weights", "yolov3.cfg")
classes = []
with open("coco (2).names", "r") as f:
    classes = [line.strip() for line in f.readlines()]

# Get output layer names
output_layer_names = net.getUnconnectedOutLayersNames()

# Function to detect cars in an image using YOLO
def detect_cars(image):
    height, width, _ = image.shape

    # Detecting objects
    blob = cv2.dnn.blobFromImage(image, 0.00392, (416, 416), (0, 0, 0), True, crop=False)
    net.setInput(blob)
    outs = net.forward(output_layer_names)

    # Showing informations on the screen
    cars_detected = 0
    for out in outs:
        for detection in out:
            scores = detection[5:]
            class_id = np.argmax(scores)
            confidence = scores[class_id]
            if confidence > 0.4 and class_id == 2:  # 2 corresponds to "car" class in COCO dataset
                # Object detected is a car with confidence > 0.5
                center_x = int(detection[0] * width)
                center_y = int(detection[1] * height)
                w = int(detection[2] * width)
                h = int(detection[3] * height)
                # Rectangle coordinates
                x = int(center_x - w / 2)
                y = int(center_y - h / 2)
                cv2.rectangle(image, (x, y), (x + w, y + h), (0, 255, 0), 2)
```

```
    while True:
        ret, frame = cap.read()
        if not ret:
            break

        # Detect cars in the frame using YOLO
        frame_with_cars, _ = detect_cars(frame)

        # Count parking spots
        num_parking_spots, frame_with_spots = count_parking_spots(frame_with_cars)

        # Calculate available parking spots
        num_cars = _
        num_available_spots = num_parking_spots - num_cars
        if num_available_spots < 0:
            num_available_spots = 0

        # Display the frame with detected cars and parking availability
        cv2.putText(frame_with_spots, f'Cars detected: {num_cars}', (10, 30), cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 255, 255), 2)
        cv2.putText(frame_with_spots, f'Available spots: {num_available_spots}', (10, 70), cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 255
        cv2.imshow('Parking Management System', frame_with_spots)

        # Break the loop if 'q' is pressed
        if cv2.waitKey(1) & 0xFF == ord('q'):
            break

    # Release the capture
    cap.release()
    cv2.destroyAllWindows()

if __name__ == "__main__":
    main()
```

```
                cars_detected += 1

    return image, cars_detected

# Function to count parking spots
def count_parking_spots(image):
    # Convert image to HSV color space
    hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)

    # Define range of green color in HSV
    lower_green = np.array([40, 40, 40])
    upper_green = np.array([80, 255, 255])

    # Threshold the HSV image to get only green colors
    mask_green = cv2.inRange(hsv, lower_green, upper_green)

    # Setup SimpleBlobDetector parameters
    params = cv2.SimpleBlobDetector_Params()

    # Filter by color
    params.filterByColor = True
    params.blobColor = 255

    # Create a detector with the parameters
    detector = cv2.SimpleBlobDetector_create(params)

    # Detect blobs
    keypoints = detector.detect(mask_green)

    # Draw detected blobs as green circles
    im_with_keypoints = cv2.drawKeypoints(image, keypoints, np.array([]), (0, 255, 0),
                                          cv2.DRAW_MATCHES_FLAGS_DRAW_RICH_KEYPOINTS)

    # Count parking spots
    parking_spots = len(keypoints)

    return parking_spots, im_with_keypoints

# Main function
def main():
    # Path to the video file
    video_path = 'Parking Spotter -- AI based Video Analytics.mp4'
    # Capture video from file
    cap = cv2.VideoCapture(video_path)
```

Output:-



Future Work

Using YOLO (You Only Look Once) for parking management has promising potential. Here are some future directions for research and development in this area:

- **Improved Object Detection**: Continuously refining object detection algorithms like YOLO to enhance accuracy, especially in crowded parking lots or in challenging lighting conditions, can significantly improve the overall performance of parking management systems.
- **Real-time Processing**: Work on optimizing YOLO or developing new algorithms to ensure real-time processing of video feeds from parking lots. This would enable instant updates on parking availability, reducing the time spent by drivers searching for parking spots.
- I**ntegration with Parking Guidance Systems**: Integrating YOLO-based parking management systems with parking guidance systems can provide drivers with real-time information about available parking spaces and guide them to the nearest vacant spot.
- **Occupancy Prediction**: Utilizing historical data and machine learning techniques, develop models to predict parking lot occupancy trends. This can help in better resource allocation and planning for parking facilities.
- **Behavioral Analysis**: Combine YOLO with other computer vision techniques to analyze driver behavior within parking lots. This could include detecting illegal parking, monitoring pedestrian movements for safety, or identifying suspicious activities for security purposes.

Conclusion:-

In conclusion, leveraging YOLOv3 for parking management systems presents a promising avenue for addressing the challenges associated with parking availability and efficiency. Through real-time object detection, YOLOv3 enables accurate identification and tracking of vehicles within parking lots, facilitating effective management and optimization of parking spaces. However, to fully realize the potential of YOLOv3 in this context, future research and development efforts should focus on enhancing object detection accuracy, optimizing real-time processing capabilities, integrating with parking guidance systems, and addressing privacy concerns. By advancing these aspects, YOLOv3-based parking management systems can significantly improve the overall parking experience for both drivers and facility operators, leading to more efficient resource utilization and enhanced urban mobility.

Work distribution

Rohan Yadav	Code
Gautam Dwivedi	Code
Muni Reddy	Report and collecting video
Upkar Chatta	Report and collecting video

References:-

(1) N. Prakash, N. Kumareshan, S. Tamilselvan and R. Gowrishankar, "Design and Management of an Intelligent Parking Slot System using Computer Vision," *2023 International Conference on Data Science, Agents & Artificial Intelligence (ICDSAAI)*, Chennai, India, 2023, pp. 1-6, doi: 10.1109/ICDSAAI59313.2023.10452432.

(2) C. -F. Hsieh, C. -Z. Lin, Z. -Z. Li and C. -H. Cho, "Automatic Vehicle License Plate Recognition Based on YOLO v4 for Smart Parking Management System," *2022 IEEE 11th Global Conference on Consumer Electronics (GCCE)*, Osaka, Japan, 2022, pp. 905-906, doi: 10.1109/GCCE56475.2022.10014165.

(3) P. Sharma, S. Gupta, P. Singh, K. Shejul and D. Reddy, "Automatic Number Plate Recognition and Parking Management," *2022 International Conference on Advances in Computing, Communication and Applied Informatics (ACCAI)*, Chennai, India, 2022, pp. 1-8, doi: 10.1109/ACCAI53970.2022.9752632.

(4) D. P. Surve, K. Shirsat and S. Annappanavar, ""Parkify" Automated Parking System Using Machine Learning," *2023 International Conference on Integrated Intelligence and Communication Systems (ICIICS)*, Kalaburagi, India, 2023, pp. 1-7, doi: 10.1109/ICIICS59993.2023.10421723.

(5) N. Gonthina, S. Katkam, R. A. Pola, R. T. Pusuluri and L. V. N. Prasad, "Parking Slot Detection Using Yolov8," *2023 3rd International Conference on Mobile Networks and Wireless Communications (ICMNWC)*, Tumkur, India, 2023, pp. 1-7, doi: 10.1109/ICMNWC60182.2023.10435799.